INTENSIONS AS SUCH: AN OUTLINE

John A. Barnden
Computer Science Department
Indiana University
Bloomington, Indiana 47405

ABSTRACT

The formal expression of propositional attitudes, especially when nested (iterated), is an important problem for AI. An interesting first-order extensional logical system for such expression has been proposed by Creary. In this system concepts (and concepts of concepts, etc.) are made explicit. The system includes "concept functions", which are special functions which act on and deliver concepts. We point out a difficulty with these functions. A alternative system is proposed, in which there is a concept-forming function corresponding roughly to complex-concept formers (especially the phrase "the proposition that") implicit in English sentences. The resulting system has a more primitive and natural notional base than Creary's has. We avoid problems with quantification inside propositions which are the objects of propositional attitudes by recasting quantified expressions into variable-free form by means of certain functions ("combinators").

I INTRODUCTION

In the past few years AI research has included some attacks on the important problem of representing beliefs, concepts, intensions, referential opacity, and the like [see e.g. (Brachman, 1979), (Hofstadter et al, 1980), (Konolige, 1982), (Maida and Shapiro, 1982), (McCarthy, 1979), (Moore, 1977), (Shapiro, 1979), (Weyrauch, 1977, 1980)]. We propose an alternative to Creary's system as described in (Creary, 1979) (which we shall call "system C"); that system is an extension of a system of (McCarthy, 1979) and is loosely based on the work of Frege (Geach and Black, 1952). Like system C, our own system keeps to first-order extensional logic, makes concepts explicit, has an infinite hierarchy of orders of concepts (concepts of concepts, concepts of concepts of concepts, etc.)*, and takes propositions to be a sort of concept. With Creary, we regard concepts as abstractions from mental entities. We claim, however, that a basic feature in system C - concept functions - is not well conceived. An example of a concept function is a function which takes a concept of a man and delivers a concept of the man's wife; another concept function might deliver the proposition that the man's wife is French. Concept functions in (Creary, 1979) are based on similar functions in McCarthy's system (McCarthy, 1979), and are akin to the "characterizing functions" in (Church, 1951, 1973, 1974). We shall claim that the use of concept functions does not allow satisfactory formalisation of nested propositional attitudes (beliefs about beliefs, beliefs about beliefs about beliefs, etc.). Our system avoids concept functions by introducing a special function which takes some concepts and delivers a complex concept similar to a definite description. Such definite-descriptional concepts include propositions, which are regarded as concepts whose extensions are truth-values (cf. the view of propositions in (Church, 1951, 1973, 1974)).

We shall use the term "denotation" *exclusively* for the relationship between expressions in English or in a logical formalism and entities in an interpretation of the system of expressions. This relationship is to be carefully distinguished from the relationship which holds between a concept and its extension (which is often also called denotation). We shall say that a concept *extends* to its extension. Thus the phrase "the concept of Mary" denotes a concept which extends to the person Mary.

We do not attempt to specify how to deduce extra beliefs that a cognitive agent holds on the basis of beliefs already ascribed to it. Such deductions are left to arise from particular axioms that some user of our formalism may choose to include in a theory. Similarly, we do not legislate about the connections between knowledge and belief — such matters are again left to the whim of the user.

Our work differs from that of (Konolige 1979) in that we avoid the casting of beliefs as expressions in some language so that statements about agents' beliefs are formalised in a metalanguage which talks about that language. The work differs from that of (Konolige 1979) and that of (Maida and Shapiro, 1982) by the inclusion of the above-mentioned concept-constructing function and concepts of it. It differs also from (Maida and Shapiro, 1982) in not insisting that items in the formalism cannot denote "extensions". The work differs from that of (Moore, 1977) in avoiding a possible-world approach to propositional attitudes.

This paper describes work done in the Department of Computer Science at Indiana University. The research was supported by the National Science Foundation under Grant MCS-8102291.

* Although there have been objections to such hierarchies, e.g. Carnap [2].

II INFORMAL CONSIDERATIONS

A. A Problem with Concept Functions

Consider the sentences

((la)) Mary is French

and

((lb)) Mike believes that Mary is French.

In system C these could be formalised as french(mary) and believe(mike, French(Mary)). Here mary and french have as their intended interpretations a particular person Mary and the predicate of being French, whereas Mary and French have as their intended interpretations a certain concept of Mary and a "whether-French-conceptrof" function whose value on a concept of a person is the propositional concept of that person being French.* We call such a function a *concept function.* The term French(Mary) denotes the proposition (a type of concept) that Mary is French.

Let us paraphrase sentence ((lb)) as

((2)) Mike believes the proposition that Mary is French.

We can view the conceptrfunction technique in system C as taking the phrase "the proposition that Mary is French" as denoting the same proposition that "the whether-French-conceptrof the concept of Mary" denotes. Thus ((lb))/((2)) is equivalent" to

((3)) Mike believes
the whether-French-concept-of the concept of Mary.

We have no objection to the equivalence of ((2)) and ((3)). The trouble arises when we go to second-order propositional attitudes, such as in the sentence

((4)) Pat believes that

Mike believes that Mary is French.

We take this to be merely an abbreviation for

((5)) Pat believes (the proposition) that
Mike believes the proposition that Mary is French.

The point is that the equivalence of ((2)) and ((3)) does not sanction an equivalence of ((5)) with

((6)) Pat believes (the proposition) that Mike believes
the whether-French-concept-of the concept of Mary

because the outer "the proposition that" sets up an opaque context. Deducing ((0)) from ((5)) is precisely the same mistake as deducing "Pat believes (the proposition) that Mike is married to the mother of Mary" from "Pat believes (the proposition) that Mike is married to the eldest sister of Jim" on the basis of an equivalence between "Mike is married to the mother of Mary" and "Mike is married to the eldest sister of Jim" (this equivalence itesIf resulting validly from an identification of Mary's mother and Jim's eldest sister). Now system C provides a formula which formalises ((6)), namely

((7)) believe(pat, Believe(Mike, French$(Mary$)))

where Mary$ and French$ are respectively an individual constant and a concept-function symbol bearing much the same relationship to Mary and French as these do to mary and french. The machinery of system C does not allow formulae which are any closer to ((5)) than ((6)) is. Unfortunately, ((5))/((4)) is precisely the sort of sentence we would like to be able to formalise, and which system C was set up to formalise.

* It appears that the symbol Mary is to be interpreted as some sort of special, "standard" concept of the person Mary, but the correctness of this appearance is not crucial to the discussion.

" This equivalence results from ordinary substitution of co-denoting expressions in a transparent context: noting that the "believe" in both ((2)) and ((3)) is extensional with respect to its object ~ it is the phrase "the proposition that" which creates an opaque context.

B. Making Proposition- Construction Explicit

Our main claim is that the satisfactory formalisation of (4))/((5)) requires explicit symbols to play the proposition-constructing role of the phrase "(the proposition) that". Our first step is to impose a further step of paraphrase: ((2)) is now regarded as a paraphrase of

((8)) Mike believes the proposition-constructed-from:
the concept of being French, and
the concept of Mary.

Here "the proposition-constructed-from" is like a two-place function. We then partially paraphrase ((5)) as

((9)) Pat believes the proposition that:
Mike believes the proposition-constructed-from:
the concept of being French, and
the concept of Mary.

The point is that the function "the proposition constructed from" itself now enters explicitly into Pat's belief, in just the way that the two-place function "the children of" enters explicitly into Pat's belief in

((10)) Pat believes that
Mike is a member of the children of
Jack and Jill.

Note that the phrase "the proposition-constructed-from" is extensional (transparent) in both its arguments.

C. Definite-Descriptional Concepts

Consider now

((H)) Mike believes that the wife of Jim is French.

Using our paraphrases, one subsidiary meaning of this is rendered as

((12)) Mike believes the proposition-constructed-from:
the concept of being French, and the concept of X

where X stands for the person who happens to be Jim's wife. ((12)) is a reading of ((H)) in which the characterization of that person as Jim's wife does not enter into Mike's belief; if Jim's wife is Mary then ((12)) is equivalent to "Mike believes that Mary is French". To capture the more direct meaning of ((11)), in which the wife characterization does enter into Mike's belief, we propose

((13)) Mike believes the proposition-constructed-from:
the concept of being French, and
(the concept-constructed-from:
the concept of the wife function, and
the concept of Jim)

where the parentheses are used to effect correct grouping. We have no general right to deduce from ((13)) that Mike believes the proposition constructed from the concept of being French and the concept of Mary. The concept c referred to by the parenthesized expression in ((13)) is (in general) a different one from the standard concept of Mary. We call c a "definite-descriptional concept". The observation now is that the concept (i.e. the proposition) that Mike believes in ((13)) is also a definite-descriptional concept. We may regard "the proposition-constructed-from" as just a convenient rewriting of "the concept-constructed-from" in certain contexts.

D. Standard Concepts and Absoluteness

For simplicity, we have been assuming and shall continue to assume that for every individual, such as Mary, there is a special concept extending to that individual and which is deemed to be the "standard" concept of the individual. A further simplifying assumption is that concepts are *absolute,* i.e. not relative to the cognitive agents which entertain them. Thus, for a given individual x, each cognitive agent has the same standard concept of x, and the function "the concept constructed from" is not parametrized by a particular cognitive agent. It is emphasized that both absoluteness and the postulation of standard concepts could be abandoned from our considerations, at the price of making the presentation of the formalism in Section III more difficult.

A most important point is that it is possible for a standard concept to be a definite-descriptional concept. For instance, the (standard) concept of Mary might actually be the definite-descriptional concept constructed from the concept of the wife function and the concept of Jim. In that case, ((13)) would be equivalent to ((12)) with 'X' replaced by 'Mary'. We do not develop such possibilities in this paper. They are not a result of our particular approach — similar things could be done in system C.

Consider now

((14)) Mike holds-in-mind the concept of Mary

(as a paraphrase of "Mike thinks about Mary"). We wonder how we would approach the formalisation of (the most direct meaning of)

((15)) Pat believes that
Mike holds-in-mind the concept of Mary.

We propose that this should be paraphrased as

((16)) Pat believes the-concept-constructed-from:
the concept of holds-in-mind,
the concept of Mike, and
(the concept constructed from:
the concept of the-concept-of,
and the concept of Mary)

(where "the concept of" is short for "the standard concept of"). This is exactly parallel to paraphrasing

((15A)) Pat believes that
Mike is-married-to the mother of Mary

as

((16A)) Pat believes the-concept-constructed-from:
the concept of is-married-to,
the concept of Mike, and
(the concept constructed from:
the concept of the-mother-of,
and the concept of Mary).

III FORMALISATION

A. Preliminaries

We replace all functions and predicates by individual constants wife, french, $, etc. which we call "functions" or "functional individuals" for convenience. We introduce the genuine function α (read "apply"), so that α(wife, jim) and α(plus, 1, 2) replace wife(jim) and plus(l, 2) respectively. The atomic formulae french(John) and meet(mike, pat) are replaced by the *terms α (*french, john)and α (meet, mike, pat). Such terms denote truth values, and the truth values TRUE and FALSE are required to be in any interpretation of a theory in our logic.

For simplicity, we abbreviate an expression $\alpha(E_0, E_1, \ldots E_n)$ to $E_0[E_1, \ldots E_n]$. Thus, for instance, α(wife, jim) is abbreviated to wife[jim].

B. Construction of Complex Concepts

In any intended interpretation of a theory in our formalism we assume there exists the concept-forming functional individuals 'the-(standard-)-concept-of and 'the-concept-constructed-from'. These are denoted by the indi-

* Technically, we need different functions a of different arities n. Also, our logic should be sorted, but for brevity we do not discuss this matter.

vidual constants $ and $ respectively. We shall often abbreviate $[x] (which is already an abbreviation) to $x when x is a single symbol. Thus a($, wife)and α($, $) can be abbreviated to $wife and $$. (The $ is inspired by Creary's notation. The ability to "apply" "functions" to themselves is commented on later.)

Let us look at some examples. We take hold-in-mind[mike, $[mary]] to denote TRUE iff Mike thinks about Mary (using the standard concept of her). If this term and the term equal [mary, wife[jim]] denote TRUE then we can deduce that hold-in-mind[mike, $[wife[jim]]] denotes TRUE. On the other hand

hold-in-mind[mike, ¢[$[wife], $[jim]]]

denotes TRUE iff Mike thinks about {the wife of Jim} . Here we use the AS subscript (meaning "as such'[1]) to indicate that the woman concerned is conceived of as the wife of Jim by Mike. This term replaces system C's hold-in-mind(mike, Wife(Jim)) where Wife denotes a concept function and Jim denotes a (standard) concept of Jim. The term

hold-in-mind[mike, ¢[$wife, ¢[$father, $jim]]]

denotes TRUE iff Mike thinks about {the wife of the father of Jim} . (Note here that we are using the $x abbreviation for $[x].) The term

hold-in-mind[mike, **¢[$[wife], $[father[jim]]]]**

denotes TRUE iff Mike thinks about {the wife of u}$_{AS}$, where u is the father of Jim. The formula is equivalent to the one derived by replacing father[jim] by bill, if indeed we have that equal [bill, father[jim]] denotes TRUE.

We have been using hold-in-mind for illustrative purposes. To get back to the question of belief, the term

believe[pat, **¢[$is-married-to, $mike, $jane]]**

denotes TRUE iff Pat believes that Mike is married to Jane (where the standard concepts of being married to, Mike and Jane are used in that belief). The term

believe[pat, **¢[$is-married-to, $mike, ¢[$mother, $jim]]]**

denotes TRUE iff Pat believes that Mike is married to the mother of Jim, where now that lady is characterized in Pat's belief as the mother of Jim.

The term

believe[mike, **¢[$equal, ¢[$tel-num-of, $jim], $1234]]**

denotes TRUE iff Mike believes that Jim's telephone number is 1234. The denotation of this term is independent of that of

believe[mike, **¢[$equal, $1234, $1234]].**

C. Iterated Propositional Attitudes

Notice the "lifting" transformation applied to

is-married-to[mike, mother[jim]]

to get the outer ¢ term in

believe[pat, **¢[$is-married-to, $mike, ¢[$mother, $jim]]].**

This transformation proceeds as follows: (i) remove all square-bracket abbreviation, by replacing every expression of form f[...] by α(f,...); (ii) replace each individual constant x by $x; and (iii) replace each expression of form α(...) by **¢[...]** It is therefore in accord with previous examples to lift ((])) to form ((2)) in:

((1)) hold-in-mind[mike, $[mary]]
((2)) believe[pat, **¢[$hold-in-mind, $mike, ¢[$$, $mary]]]**

((2)) denotes TRUE iff Pat believes the proposition that Mike thinks about Mary, where Mike is supposed by Pat to use her standard concept. Similarly, the term

((3)) believe[pat,
¢[$hold-in-mind, $mike,
¢[$¢, ¢[$$, $wife], ¢[$$, $jim]]]]

denotes TRUE iff Pat believes that Mike thinks about the wife of Jim, where Mike is supposed by Pat to use the characterization of her as the wife of Jim. ((3)) is derived from hold-in-mind[mike, **¢[$wife, $jim]]** by "lifting". ((3)) can be read as "Pat believes that: {Mike holds-m-in nd the-conceptrconstructed-from the concept of wife-of and the concept of Jim}$_{AS}$".

The reader is invited to try writing a formula for a third-order propositional attitude ("George believes that Pat believes that Mike believes that ...."). We get terms whose length increases exponentially with number of lifting steps. However, if we introduce for each constant symbol x the abbreviations **¢:$x, ¢2:$x, ¢3:$x,** etc., where: **¢:$x** stands for the lifted version of $[x] (i.e. **¢[$$, $x]); ¢2:$x** stands for the lifted version of **¢:$x** (i.e. **¢[$¢, ¢[$$, $$], ¢[$$, $x]])**; and so on, then we get a approximately linear increase in the size of terms.

We introduce a special functional individual "ext-of" which delivers the extension of a concept. So, for instance, **ext-of[$[mary]]** denotes Mary, and ext-of[$[wife[jim]]] and **ext-of[¢[$wife,** $jim]] both denote Jim's wife. The term **ext-of[¢[$**french, Smary]] denotes whatever truth-value french[mary] denotes. We assume that there are axiom schemata*

equal[ext-of[$[x]], x]

and

• To say that these terms are axiom schemata is to say that for an interpretation of a theory in our logical system to be a model it must make these terms denote TRUE. Also, we assume that equal is always interpreted in a special way and that suitable axioms for it are provided.

equal[ext-of[¢[E_0, ..., E_n]],

(ext-of[E_0])[ext-of[E_1], ..., ext-of[E_n]].

D. Connectives

The first part of our treatment of connectives is to cast them as "functional" constants whose only distinctive quality is that in any interpretation they must denote certain truth-value-returning "functions". For instance, the term **and[hit[john, judy], hit[judy, john]]** denotes TRUE iff both **hit[john, judy]** and **hit[judy, john]** denote TRUE. We assume the existence of "truth-functional" constants **not**, **and**, **or** and **implies**. We allow **and** and **or** to have an arbitrary number of arguments.

The treatment of the inclusion of connectives in propositions which are the objects of propositional attitudes now falls out immediately from our preceding considerations. For example, we could have

believe[mike,
¢[$and, ¢[$hit, $john, $judy], ¢[$hit, $judy, $john]]]

by lifting the **and[...]** term above.

E. Quantifiers

To express quantification inside propositions which are themselves the objects of propositional attitudes, (Creary, 1979) uses special variable-binding operators, e.g. **Exist**. (These operators replace a somewhat cumbersome technique used by (McCarthy, 1979).) In contrast, our treatment avoids variables in the first place, and therefore avoids the need for special variable-binding operators. The avoidance arises from two hypothetical moves: 1) casting quantification temporarily in terms of functional abstraction (λ-abstraction), and 2) casting functional abstraction into variable-free form by means of special "functions" called combinators. The first step is essentially a technique used in (Church, 1951). We regard

($\forall$x)(implies[man[x], mortal[x]])

as an abbreviation of

all[λx. implies[man[x], mortal[x]]].

Here the λ-expression is a term denoting a "functional" individual. The **all** is a truth-valued "function" which, like a connective, must be given a special meaning in any interpretation. That is, **all[f]** for any one-place truth-valued "function" f denotes TRUE iff for each x in the domain of interpretation the "application" of f to x yields TRUE. Similarly, **($\exists$x)(or[man[x], woman[x]])** is abbreviated to **some[λx. or[man[x], woman[x]]].**

The second step takes λ-expressions to be rewritings of terms which use combinators (certain "functional" constants) but contain no variables. The combinators we use are **I** (the identity "function") and denumerably many symbols π^{β} where superscript β is a non-empty slash-separated list of (possibly empty) comma-separated lists of non-negative integers (e.g. (1,2)/2/(1,4,6); or (0,2); or ()). If β has k components then π^{β} denotes, so to speak, λ-abstraction of a certain mode: each element of the list β corresponds to a bound variable in a λ-abstraction, and every integer within that element specifies the positions in which the bound variable is used in the body of the λ-abstraction.

In detail, if β has one element then (π^{β}[E_0, ..., E_n]) [x] equals F_0[F_1, ..., F_n] where: F_i is E_i[x] if i is in the list of integers which is the only element of β, and is otherwise just E_i. As an example, the expression **λx. implies[human[x], mortal[x]]** is considered merely a re-notation of $\pi^{(1,2)}$**[implies, human, mortal]**.

If β has k > 1 elements, then
(π^{β}[E_0, ..., E_n]) [x_1, ..., x_k]
equals ($\pi^{\beta'}$[F_0, ..., F_n]) [a_2, ..., a_k]
where: F_i is E_i[a_1] if i is in the first list of integers in β, and is otherwise just E_i; and β' is the tail of β. For example, the expression λx.λy. or[p[x], q[x,y]] is a re-notation of $\pi^{(1,2)/(2)}$**[or, p, $\pi^{(1)/(2)}$[q, I, I]]**.

The point now is that we can easily "lift" expressions involving quantification up by one intensional level, because we have replaced all conventional quantificational constructs by ordinary applicative expressions. The only new factor is that we have the new "functions" π^{β}, **some** and **all**. For instance, we can obtain

believe[mike, ¢[$all, ¢[$$\pi^{(1,2)}$, $implies, $man, $mortal]]],

where the belief deploys the standard concepts of universal quantification, of implication, of being human, of being mortal, and of "λ-abstraction in mode β".*

---

* The form of "function" abstraction we have introduced via the π^{β} combinators allows the definition of self-contradictory "functions", e.g. $\pi^{(1)}$**[If, $\pi^{(1)}$[equal, $\pi^{(0,1)}$[I, I], 0], 1, 0]** where **If[true, x, y]** equals **x** and **If[false, x, y]** equals **y**. We get a contradiction from applying this "function" to itself. Something must be done to avoid this classic problem, and several approaches are being considered. One is to use a type hierarchy similar to that in (Church, 1951, 1973, 1974). However, the one which is receiving closest attention is based on the relativization device used in Zermelo-Fraenkel set theory to avoid the Russell paradox (which is akin to the self-application paradox just mentioned). That is, any set definition must be made relative to some set A, as in {x ∈ A | ... } Let us regard "functions" as tuple-sets in the domain of interpretation, so that in particular a π^{β}[...] term denotes a set of tuples. (We depart from standard set theory in allowing cycles in the set-membership relation, so that self-application of "functions" is possible.) Instead of regarding π^{β}[...] terms as re-notations of λ-expressions, we now regard them as re-notations of set-definition expressions of form {(x,y,...z) ∈ P | <some condition on x,y,...z> }. Here P denotes some set of tuples. The replacement of λ-expressions by such relativized set-definition expressions serves the purpose of avoiding inconsistent function definitions. A π^{β}[...] term must now contain a tuple-set-valued term (possibly a "function" term) P as its zeroth argument. In the following, we will ignore these modifications.

F. Some Examples

We may formalise "Pat knows Mike's telephone number" by

(∃n) know[pat, ¢[$equal, ¢[$tel-num, $mike], $n]]

which is a rewriting of

some[$\pi^{(2)}$[know, pat,
$\pi^{(3)}$[¢, $equal, ¢[$tel-num, $mike], $]]].

(We might also have used a formalisation in terms of correct belief.) We could formalise "Pat wonders what Mike's telephone number is" (adapted from (Maida and Shapiro, 1982)) by

curious[pat, ¢[$tel-num, $mike]].

(where we assume **curious** denotes the "predicate" of wondering what the extension of some concept is.) Similarly, "Mary wonders whether John is taller than Bill" (adapted from (Maida and Shapiro 1982)) could be formalised by

curious[mary, ¢[$taller, $john, $bill]].

In (Creary, 1979) Creary gives three formulae corresponding to different readings of "Pat believes Mike believes Jim's wife is French". We have three corresponding terms, which, however, do not mean *exactly* the same thing. The differences arise from the formulation in terms of ¢ rather than concept functions.

The first term is

((1)) believe[pat, ¢[$believe, $mike,
¢[$¢, ¢:$french, ¢[$¢, ¢:$wife, ¢:$jim]]]]

where ¢:$x is an abbreviation for ¢[$$,$x] and where the outer ¢[...] is the lifted form of

believe[mike, ¢[$french, ¢[$wife, $jim]]].

((1)) means that Pat believes that Mike believes that Jim's wife is French, where Pat's belief involves the characterization of (Mike's) characterization: wife of Jim. ((1)) expresses what is arguably the most natural reading of the above sentence.

The second term is

((2)) (∃ C',C)(and[believe[pat,
¢[$believe, $mike, ¢[$¢, ¢:$french, C']]],
concept-of[C', C], concept-of[C, wife[jim]]])

(transformed into combinator form) where now the concept c of Jim's wife which is imputed to Mike is merely *some* concept of her (c could be the denotation of ¢[$mother, $sally] for instance), and the concept c' used by Pat is simply *some* concept of that concept c (e.g. the one denoted by $[¢[$mother, $sally]]).

The third term is

((3)) believe[pat,
¢[$some,
¢[$$\pi^{(1,2)}$, $and, ¢[$$\pi^{(2)}$, $believe, $mike,
¢[$$\pi^{(3)}$, $¢, ¢:$french, $I]],
¢[$$\pi^{(1)}$, $concept-of, $I, ¢[$wife, $jim]]]]]

where the outer ¢[...] is the lifted form of

(∃ C)
{and[believe[mike, ¢[$french, C]],
concept-of[C, wife[jim]]]}

transformed first into the combinator form:

some[$\pi^{(1,2)}$[and, $\pi^{(2)}$[believe, mike, $\pi^{(3)}$[¢, $french, I]],
$\pi^{(1)}$[concept-of, I, wife[jim]]]].

((3)) means that Pat believes that Mike believes that Jim's wife is French, where part of Pat's belief is the imputation to Mike of *some* concept of the person characterized in Pat's belief as Jim's wife.

Consider the Creary formula ((7)) and sentences ((5, 6)) of Section II.A again; formula ((7)) formalises sentence ((6)), which we contrasted with ((5)). Our formulations of ((5)) and ((6)) are:

((4)) believe[pat, ¢[$believe, $mike,
¢[$¢, ¢[$$, $french], ¢[$$, $mary]]]]

((5)) believe[pat, ¢[$believe, $mike,
¢[¢[$$, $french], ¢[$$, $mary]]]]

where the outer ¢[...] is the lifted form of

believe[mike, ($[french])($[mary])]]

so that we are paraphrasing the phrase "the whether-French-concept-of" in ((6)) of II.A as "the concept of being French". We are also now postulating that the terms $french and ¢[$$,$french] denote "functions".

We observe that if we adopt the axiom schema

equal[¢[E_0, ..., E_n], E_0[E_1, ..., E_n]]

then ((5)) reduces to

believe[pat, $believe[$mike, (($$)($french))[($$)($mary)]]]

which is analogous to the formula ((7)) of II.A. Our reconstruction of terms closely analogous to those of system C by means of the above axiom schema serves the purpose of explicating the notional basis of that system.

IV CONCLUDING REMARKS

It is hoped that the discussion of Section II, which achieves more formal explication later in the paper, shows that the $ function is at least as natural as concept functions are. In particular, the use of $ results in formulae closer in form to natural-language sentences than the formulae of system C do, in that * often corresponds to the phrase "the proposition that'[1] and therefore allows that phrase itself to be an explicit component of a conceptual characterization at a higher intensional level.

A number of semantic-network researchers [e.g. (Brachman, 1979), (Shapiro, 1979), (Maida and Shapiro, 1982)] have realized the importance of basing their formalisms on intensions (concepts, descriptions, propositions, ...) rather than on "extensions". Now, semantic network formalisms are very similar to (and often just re-expressions of) logical systems. If our analyses in this paper are correct, therefore, we must suspect that existing semantic network formalisms need some modification, in order to bring in entities corresponding to $ and $.

It is important to note that, although we have called $ a concept constructor, the concepts which are constructed are not necessarily to be regarded as expressions or something similar to expressions. (In fact, we make no claim about the particular psychological nature of constructed concepts. All we say is that constructed concepts are transparently determined by the concepts they are constructed from.) It would be possible in a theory in our formalism to have two $[...] terms denoting the same concept even though the arguments in the terms denote different concepts. For instance, it would be possible to have $$odd, $x] denoting the same concept as $[$not, $[$even, $x]] for every numerical term x. It is up to individual users of the formalism to decide whether such identities are appropriate. Similar points apply to Creary's formalism, but it is not clear whether he intends to allow such identities.

There is no claim that the system as sketched here is the last word on the issues dealt with. For instance, there is the question of whether it is plausible that beliefs should be couched in terms of the concepts denoted by the $w* symbols (though we feel that our treatment of quantification is at least as plausible as those of Creary and McCarthy). Note that a person's entertaining the concept denoted by a *[$w*, ...] term does not imply that the person actually entertains the concepts denoted by the arguments to the *[...] term, in particular the $B term.

A fuller paper which discusses in more detail the general and technical issues raised by our considerations in this report is in preparation.

ACKNOWLEDGMENTS

1 should like to thank Mitchell Wand for useful comments on an initial draft of the paper. I am indebted to Robert Filman, who was the original director of the grant which supported this research.

REFERENCES

[1| Brachman, R.J. On the epistemological status of semantic networks. In N.V. Findler (ed), Associative Networks, Academic Press: New York, 1979.

(2) Carnap, R. Meaning and Necessity, 2nd. Ed., University of Chicago Press: Chicago, 1956.

[3| Church, A. A formulation of the logic of sense and denotation. In P. Henle (ed), Structure, Method and Meaning: Essays in Honor of Henry M. Shefftr, Liberal Arts Press: New York, 1951.

[4] Church, A. Outline of a revised formulation of the logic of sense and denotation (part I). Nous 7(1973), 24- 33.

[5] Church, A. Outline of a revised formulation of the logic of sense and denotation (part II). Nous 8 (1974), 135- 156.

[6] Creary, L.G. Propositional attitudes Fregean representation and simulative reasoning. In Procs. 6th. Int. Joint Conf. on Artificial Intelligence, Tokyo, 1979.

[7] Curry, H.B. and Feys, R. Combinatory Logic, North-Holland: Amsterdam, 1958.

(8| Geach, P. and Black, M. (eds) Translations from the Philosophical Writings of Gottlob Frege, Blackwell: Oxford, 1952.

[9] Hofstadter, D.R., Clossman, G.A., and Meredith, M.J. Shakespeare's plays weren't written by him, but by someone else of the same name. Tech. Rep. 96, Computer Science Dept., Indiana Univ., July 1980.

[10] Kalish, D. and Montague, R. 'That'. In R.H. Thomason (ed.), Selected Papers of Richard Montague, Yale University Press: New Haven, 1974.

[11) Konolige, K. A first-order formalisation of knowledge and action for a multi-agent planning system. In J.E. Hayes, D. Michie and Y-H Pao (eds), Machine Intelligence 10, Ellis Horwood: Chichester, 1982.

[12) Maida, AS. and Shapiro, S.C. Intensional concepts in propositional semantic networks. Cognitive Science 6, 4 (1982), 291-330.

[13] McCarthy, J. First order theories of individual concepts and propositions. In J.E. Hayes, D. Michie and L.I. Mikulich (eds), Machine Intelligence 9, Ellis Horwood: Chichester, 1979.

(14) Moore, R.C. Reasoning about knowledge and action. In Procs. 5th. Int. Joint Conf. on Artificial Intelligence, MIT, 1977.

[15] Shapiro, S.C. The SNePS semantic network processing system. In N.V. Findler (ed.), Associative Networks, Academic Press: New York, 1979.

[16| Weyrauch, R.W. A User's Manual for VOL, Artificial Intelligence Laboratory Memo AIM-235.1, Stanford University, July 1977.

[17] Weyrauch, R.W. Prolegomena to a theory of mechanized formal reasoning. Artificial Intelligence 13, 1 (1980), 133- 170.